

# تحلیل باج افزار ویندوزی Speedy

تاریخ گزارش: ۳۰ اردیبهشت ۱۳۹۹

## مقدمه

به تازگی آزمایشگاه امنیت کی پاد بدافزاری برای پلتفرم ویندوز در زیرساخت سطح کاربران این سیستم عامل در کشور جمهوری اسلامی ایران رصد و شناسایی کرده است که این بدافزار قابلیت های ویژه و منحصر بفردی دارد. شایان ذکر است، در هنگام بررسی اولیه که توسط آزمایشگاه امنیت کی پاد صورت گرفت، این بدافزار توسط ضدباج افزار کی پاد به صورت کامل قابل شناسایی بوده است.

شایان ذکر است، باج افزارهای مشابه با این باج افزار با نام هایی از قبیل STOP، DJVU، Drume، StopData و تاکنون مشاهده شده است که تقریباً یک ساختار و عملکرد مشابه با یکدیگر دارند و فرمت فایل mpal، mogranos، krusop، lotej، nvetud، cosakos، prandel، kovasoh، zatrov، masok، brusaf، londec، یا krusop تنها بخشی از فرمت فایل هایی هستند که این بدافزار استفاده می کند.

شایان ذکر است، این باج افزار مانند انواع دیگر از باج افزارهای خانواده STOP/DJVU از طریق هرزنامه ها و فیشینگ گسترش پیدا می کند. پس از آلوده شدن سیستم قربانی، این باج افزار از الگوریتم های بر پایه AES-256 و RSA یا یک الگوریتم جدیدتر برای رمزگذاری فایل های میزبان استفاده می کند. سرعت عملکرد این باج افزار آنقدر بالاست که به محض آلوده شدن سیستم، قربانی می تواند تغییر پسوند فایل هایش به mpal یا opqz و دیگر پسوندها را مشاهده کند. در ادامه باج افزار فایل های پشتیبانی سایه (Shadow) قربانی را نیز حذف می کند تا احتمال بازگردانی فایل ها غیرممکن شود.

هنوز آماری در رابطه با تعداد قربانیان این باج افزار در ایران منتشر نشده است اما قربانیان می گویند هرکس مبالغی بین ۳۵۰ تا ۸۰۰ دلار را از آن ها به عنوان باج درخواست کرده اند. شروع گسترده آلودگی سیستم ها به باج افزارهای این گروه از سال ۲۰۱۸ بود و از این رو گروه تصمیم گرفت نسخه های مختلف و پیچیده تر آن را با پسوندهای مختلف و البته الگوریتم های پیچیده تر منتشر کند.

به هر صورت، در حال حاضر باج افزارها تهدید جدی به شمار می روند، و هر از گاهی قربانیان زیادی از کسب و کارهای کوچک تا شرکت های تجاری بسیار بزرگ می گیرند. باج افزاری که در ادامه به تحلیل آن خواهیم پرداخت تاکنون توانسته است برخی از سازمان های رسمی کشور را مورد هدف قرار بدهد و با موفقیت آن ها را آلوده کند.

# مشخصات جاسوس افزار

در جدول زیر، مشخصات کلی باج افزار ویندوزی Speedy به صورت خلاصه آورده شده است. در ادامه، تحلیل این بدافزار که از خانواده باج افزارها به شمار می رود، با جزئیات دقیق تری آورده شده است.

e93a3705fdd01cf942e4830e148f0b66	شناسه MD5
4F640DA558A758DD5D58483C150E6D41A05F3766	شناسه SHA-1
f6eb4d6bdb959f3ac087e153db2077a0372d3ed984f32184ff04d273ffdfc64c	شناسه SHA-256
سیستم عامل ویندوز (Windows-based OS)	پلتفرم هدف
باج افزار «Ransomware»	نوع بدافزار
ندارد	یارا
ندارد	اسنورت
این بدافزار که توسط آزمایشگاه امنیت کی پاد «Speedy باج افزار» نامگذاری شده است، که توسط ضد باج افزار کی پاد قابل شناسایی است.	
توضیحات بدافزار	

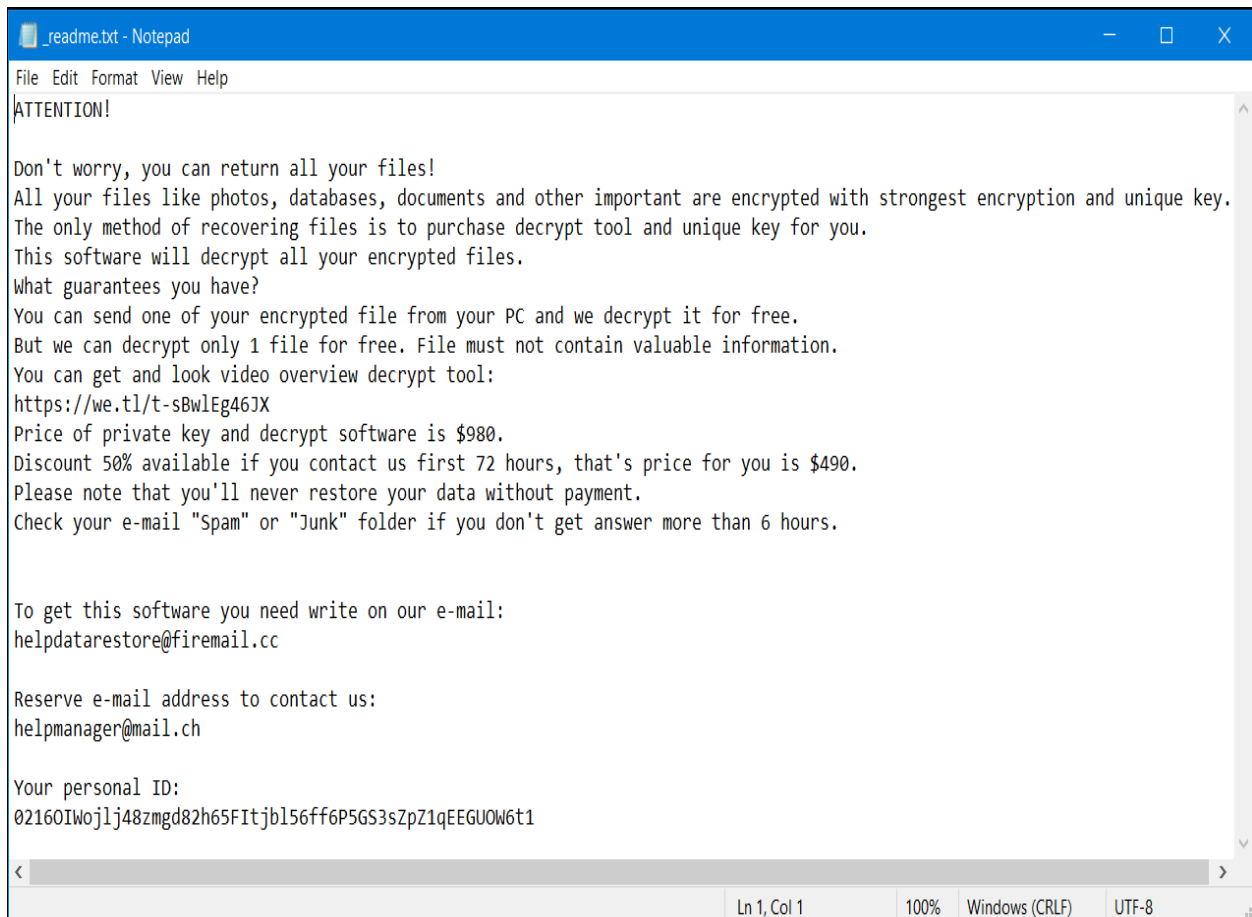
## فهرست

۱.....	مقدمه
۲.....	مشخصات جاسوس افزار
۴.....	مقدمه باج افزار Speedy
۵.....	تجزیه و تحلیل بدافزار Speedy
۶.....	اطلاعات فایل اجرایی بدافزار
۷.....	تکنیک Module Stomping
۱۱.....	تحلیل کد تزریق شده به حافظه هیپ
۱۹.....	نتیجه گیری
۱۹.....	نشانه نفوذگر «IOC»

# مقدمه باج افزار Speedy

این نسخه باج افزار که با عنوان Speedy نامگذاری شده است، بعد از اینکه عملیات خود را آغاز می کند، تمامی فایل ها را با فرمت رمزنگاری خواهد کرد. البته Speedy یکی از باج افزارهای عملیاتی از خانواده STOP/DJVVU است. همچنین در هر پوشه یک فایل \_readme.txt ایجاد می کند که حاوی راهنما برای پرداخت باج و بازیابی فایل های رمزنگاری شده است.

با توجه به متن موجود در این فایل راهنما، مبلغ باج درخواستی ۹۸۰ دلار است، اما اگر در ۷۲ ساعت ابتدایی مبلغ باج را پرداخت کنید، ۵۰ درصد تخفیف به شما تعلق خواهد گرفت. با این حال، تاکنون گزارشی از دریافت کلید رمزگشایی فایل ها بعد پرداخت باج وجود ندارد. همچنین هیچگاه توصیه نمی شود که مبلغ باج توسط قربانیان پرداخت شود چون به آن ها فرصت می دهد که مجدد شما را مورد حمله قرار بدهند.



```
_readme.txt - Notepad
File Edit Format View Help
ATTENTION!

Don't worry, you can return all your files!
All your files like photos, databases, documents and other important are encrypted with strongest encryption and unique key.
The only method of recovering files is to purchase decrypt tool and unique key for you.
This software will decrypt all your encrypted files.
What guarantees you have?
You can send one of your encrypted file from your PC and we decrypt it for free.
But we can decrypt only 1 file for free. File must not contain valuable information.
You can get and look video overview decrypt tool:
https://we.tl/t-sBwlEg46jX
Price of private key and decrypt software is $980.
Discount 50% available if you contact us first 72 hours, that's price for you is $490.
Please note that you'll never restore your data without payment.
Check your e-mail "Spam" or "Junk" folder if you don't get answer more than 6 hours.

To get this software you need write on our e-mail:
helpdatarestore@firemail.cc

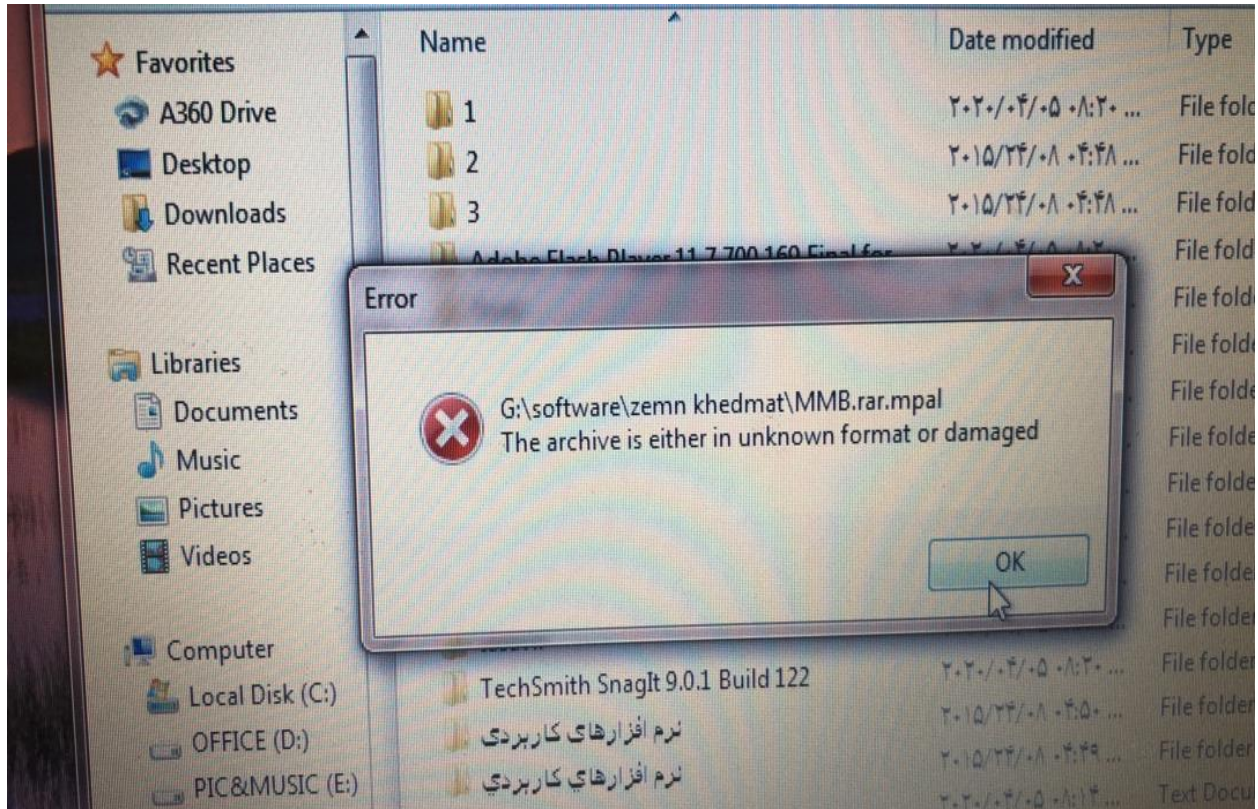
Reserve e-mail address to contact us:
helpmanager@mail.ch

Your personal ID:
02160Iwojlj48zmgd82h65FItjbl56ff6P5GS3sZpZ1qEEGUOW6t1

Ln 1, Col 1 100% Windows (CRLF) UTF-8
```

تصویر ۱: متن باج خواهی باج افزار Speedy

البته نمونه دیگری از فعالیت‌های مشابه باج‌افزارهای خانواده STOP/DJVU در موقعیت‌های دیگر هم مشاهده شده است، به عنوان مثال در تصویر ۲ وضعیت کامپیوتر یک قربانی را مشاهده می‌کنید که توسط باج‌افزاری از خانواده STOP/DJVU مورد هدف قرار گرفته است و تمامی فایل‌های مهم او با فرمت .mpal. رمزنگاری شده است.



تصویر ۲: آلودگی ماشینی یک کاربر به باج‌افزار

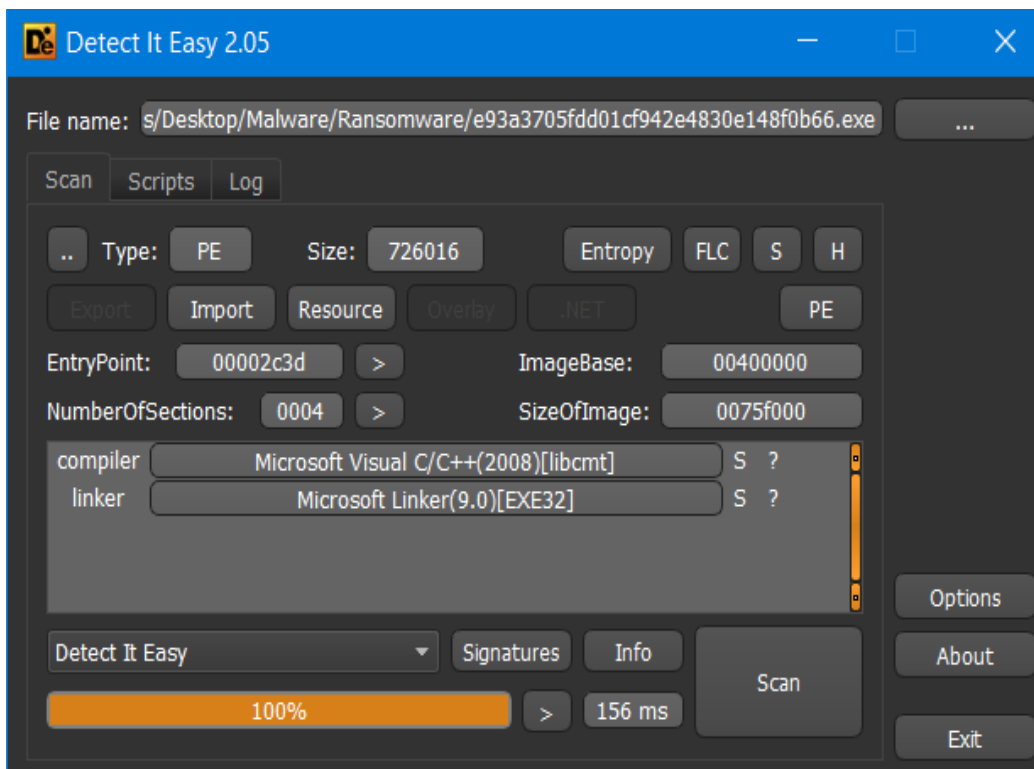
به هر صورت، از این خانواده بدافزارهای مشابه با نام‌هایی از قبیل STOP، Puma، DJVU، Drume، StopData و تاکنون مشاهده شده است که تقریباً یک ساختار و عملکرد مشابه با یکدیگر دارند و فرمت فایل .mpal، .opqz، .londec، .brusaf، .masok، .zatrov، .prandel، .kovasoh، .lotej، .nvetud، .cosakos، .mogranos یا .krusop تنها بخشی از فرمت‌هایی هستند که این بدافزار استفاده می‌کند.

## تجزیه و تحلیل بدافزار Speedy

همانطور که در قسمت بالا ذکر شد، خانواده باج‌افزارهای STOP/DJVU با نام‌هایی از قبیل STOP و DJVU و ... تاکنون مشاهده شده‌اند که تقریباً یک ساختار و عملکرد مشابه با یکدیگر دارند و فرمت فایل .mpal یا .opqz فرمت فایل‌هایی هستند که این بدافزار حداقل برای هدف قرار دادن سازمان‌ها و کاربران ایرانی تاکنون استفاده کرده است.

## اطلاعات فایل اجرایی بدافزار

نمونه باج‌افزاری که از این خانواده قرار است، مورد بررسی و تحلیل قرار بدهیم، مانند نمونه‌های پیشین با زبان CPP و کامپایلر MSVC نوشته و ایجاد شده است که در تصویر ۳ این اطلاعات را می‌توانید مشاهده کنید. شایان ذکر است، شناسه MD5 این بدافزار پیش از این در جدول معرفی آن ارائه شده است که می‌توانید از آن برای دنبال کردن فعالیت‌های این بدافزار یا مشاهده دیگر گزارش‌های مرتبط با این بدافزار استفاده کنید.



تصویر ۳: اطلاعات فایل PE بدافزار

هنگامیکه دراپر این بدافزار اجرا می‌شود، در گام اول تابع `GetThickCount` را فراخوانی می‌کند که کاربرد آن با محوریت ضددیباگ است. به عبارت دیگر، توسعه‌دهندگان بدافزار از آن استفاده می‌کنند، تا اجازه تحلیل بدافزار را به تحلیل‌گران و یا سندباکس‌ها ندهند. از همین روی، وقتی فایل اجرایی دراپر باج‌افزار را تحلیل و بررسی می‌کنیم، در گام اول فراخوانی این تابع را مشاهده خواهیم کرد.

در تصویر ۴، فراخوانی این تابع نمایش داده شده است. بعد از اینکه این تابع فراخوانی شود، مقدار زمانی که بازگشت می‌دهد، مدت زمانی است که ماشین عملیاتی بوده است. در نتیجه بدافزار به سادگی با بررسی این عدد بازگشتی می‌تواند تشخیص بدهد که بر روی چه نوع سامانه‌ای قرار گرفته است. اگر این مدت زمان کم باشد، فعالیت خود را ملقی می‌کند، چون احتمال وجود سندباکس در این شرایط وجود دارد.

```
Decompile: FUN_00401430 - (e93a3705fdd01cf942e4830e148f0b66.exe)
83
84 local_8 = DAT_004a800c ^ (uint)&stack0xffffffff;
85 sVar1 = _strlen((char *)&lpString_004aa1f0);
86 if (sVar1 == 0x4a090) {
87     FUN_004020d0(0x80000000,0x41d78fce,0x20000000,0x41dd92c4);
88     _fclose((FILE *)0x0);
89     FUN_00402390(0);
90     FUN_004020f0(0,0);
91     FUN_004026c0();
92 }
93 local_c18 = 0x103a7e7;
94 do {
95     GetTickCount();
96     if (DAT_00b52a74 == 0x13bc) {
97         FindNextVolumeW((HANDLE)0x0,local_83c,0);
98         FindFirstVolumeMountPointW((LPCWSTR)0x0,(LPWSTR)0x0,0);
99         ProcessIdToSessionId(0,(DWORD *)0x0);
100    }
101    local_c18 = local_c18 - 1;
102 } while (local_c18 != 0);
103 iVar2 = lstrlenA((LPCSTR)&lpString_004aa1f0);
104 if (iVar2 == 0x161a1) {
105     iVar2 = 99;
106     puVar3 = local_b1c;
107     do {
108         puVar3[1] = 0xf;
109         *puVar3 = 0;
110         *(undefined *)(puVar3 + -4) = 0;
111         puVar3 = puVar3 + 7;
112         iVar2 = iVar2 + -1;
113     } while (-1 < iVar2);
114     FUN_00402950();

```

### تصویر ۴: فراخوانی تابع GetThickCount

بعد از اینکه تابع مذکور فراخوانی شود، روتینی اجرا خواهد شد که باینری دیگری را به درون حافظه هیپ تزریق خواهد کرد و در نهایت با استفاده از تکنیک Module Stomping آن را در حافظه به صورت On the fly اجرا خواهد کرد.

## تکنیک Module Stomping

یکی از تکنیک‌هایی که در عموم باج‌افزارهای این خانواده مشاهده می‌شود، استفاده از تکنیک Module Stomping است که به مهاجم اجازه می‌دهد کدی (مثلا یک فایل باینری) را در فضای هیپ تزریق و در نهایت اجرا کند، که در حال عادی این کار ممکن نیست چون فضای هیپ فقط خواندنی (Read-Only و Writeable) است.

در این نسخه از باج‌افزار با مشاهده فرخوانی توابع سیستمی ویندوز از قبیل VirtualProtect و VirtualAlloc می‌توان حدس زد که توسعه‌دهندگان این بدافزار مجدد از این تکنیک استفاده کرده‌اند تا کدی را به صورت On the fly بر روی حافظه اجرا کنند. در تصویر ۵، استفاده از این توابع توسط باج‌افزار مذکور نمایش داده شده است.

شایان ذکر است، توابع VirtualAlloc، VirtualProtect و VirtualFree در این لیست برای ما اهمیت فراوانی دارند، زیرا می‌توانیم با تنظیم یک Breakpoint بر روی آن‌ها تشخیص بدهیم که وقتی بدافزار از این رابط‌های برنامه‌نویسی ویندوز را فراخوانی می‌کند، چه هدفی در پس‌زمینه وجود دارد.

Address	Ordinal	Name	Library
004A30F4		SetFilePointer	KERNEL32
004A30E4		SetHandleCount	KERNEL32
004A3134		SetLastError	KERNEL32
004A3168		SetStdHandle	KERNEL32
004A30CC		SetUnhandledExceptionFilter	KERNEL32
004A30AC		SetupComm	KERNEL32
004A3040		SizeofResource	KERNEL32
004A3100		Sleep	KERNEL32
004A30C0		TerminateProcess	KERNEL32
004A3124		TlsAlloc	KERNEL32
004A312C		TlsFree	KERNEL32
004A3120		TlsGetValue	KERNEL32
004A3128		TlsSetValue	KERNEL32
004A30C8		UnhandledExceptionFilter	KERNEL32
004A3164		VirtualAlloc	KERNEL32
004A3144		VirtualFree	KERNEL32
004A3098		VirtualProtect	KERNEL32
004A3090		WTSGetActiveConsoleSessionId	KERNEL32
004A303C		WideCharToMultiByte	KERNEL32
004A319C		WriteConsoleA	KERNEL32
004A31A4		WriteConsoleW	KERNEL32
004A310C		WriteFile	KERNEL32
004A3014		_write	KERNEL32
004A3054		IstrcatA	KERNEL32
004A3060		IstrlenA	KERNEL32

تصویر ۵: توابع وارد شده به فایل اجرایی

با بررسی این توابع می‌توانیم سوال‌های گوناگونی را پاسخ بدهیم از قبیل اینکه آیا قرار است کدی توسط بدافزار به درون حافظه هیپ تزریق و اجرا شود؟ اگر چنین است، آن کد چه کاری انجام می‌دهد و چه ساختاری دارد؟ با تحلیل و به دست



آوردن این دست اطلاعات بهتر می‌توانیم با ساختار داخلی این بدافزار آشنا شویم و در نهایت برای آن یک راه‌حل پایدار دفاعی ارائه کنیم.

در تصویر ۶، فرخوانی تابع VirtualAlloc را مشاهده می‌کنید که بعد از فراخوانی آن قسمتی از حافظه هیپ به بدافزار تخصیص داده خواهد شد که می‌توان در آن کدی را تزریق کرد و در نهایت با فراخوانی تابع VirtualProtect به آن سطح دسترسی Executable داد.

پس از اینکه با موفقیت سطح دسترسی Executable به آن صفحه از حافظه (Memory Page) ارائه شود، بدافزار می‌تواند کد قرار گرفته در آن بلاک از حافظه را به صورت On the Fly بر روی حافظه اجرا کند. در تصویر ۵، فرخوانی VirtualAlloc و ناحیه‌ای از حافظه هیپ که بعد از فراخوانی این رابط برنامه‌نویسی به بدافزار تخصیص داده شده است، مشاهده می‌کنید.

### تصویر ۶: فراخوانی VirtualAlloc و حافظه تخصیص از آن

بعد از فراخوانی VirtualAlloc، حافظه هیپ درخواستی به بدافزار تخصیص داده خواهد شد. بعد از فراخوانی این رابط برنامه‌نویسی، آدرسی که در رجیستر EAX ذخیره می‌شود، آغاز حافظه تخصیص داده شده به بدافزار را نمایش می‌دهد. شایان ذکر است، اگر به این آدرس در محیط Dump رجوع کنید، محتویات این حافظه تخصیص داده شده را مشاهده خواهید کرد که با مقادیر NULL یا 00 پر شده‌اند.

در گام بعد وقتی تکه کد مورد نظر توسعه‌دهندگان بدافزار در این قسمت از حافظه قرار گرفت، تابع VirtualProtect را فراخوانی خواهند کرد تا به آن ناحیه از حافظه هیپ سطح دسترسی PAGE\_EXECUTE\_READWRITE بدهند که در نهایت بتوان در آن ناحیه کدی را اجرا کرد (در اینجا یک باینری است که عمل رمزنگاری فایل‌ها را بر عهده دارد). در تصویر ۷، پارامترهایی عبوری به VirtualProtect نمایش داده شده است.

004013B4	FF15 AC304A00	call dword ptr ds:[<&SetupComm>]	
004013BA	8B15 742AB500	mov edx,dword ptr ds:[B52A74]	
004013C0	A1 E828B500	mov eax,dword ptr ds:[B528E8]	
004013C5	8D8D F8F7FFFF	lea ecx,dword ptr ss:[ebp-808]	
004013CB	51	push ecx	
004013CC	6A 40	push 40	A pointer to a variable that receives the previous access PAGE_EXECUTE_READWRITE
004013CE	52	push edx	
004013CF	50	push eax	
004013D0	FF15 98304A00	call dword ptr ds:[<&VirtualProtect>]	
004013D6	8B4D FC	mov ecx,dword ptr ss:[ebp-4]	
004013D9	33CD	xor ecx,ebp	
004013DB	E8 D80C0000	call e93a3705fdd01cf942e4830e148f0b66.4	

تصویر ۷: پارامترهای عبوری به رابط VirtualProtect

بعد از اینکه VirtualAlloc و VirtualProtect با موفقیت کار خود را انجام بدهند، در حافظه هیپ تخصیص داده شده توسط VirtualAlloc، کدی را که توسط بدافزار در ادامه اجرا خواهد شد، خواهیم دید. در ادامه می‌توانیم با دامپ گرفتن از آن کد به صورت مستقل در قالب یک فایل باینری مجزای دیگر تحلیل آن را آغاز کنیم.

همانطور که در تصویر ۸ در قسمت پنجره Dump مشاهده می‌کنید، کدی که به درون حافظه هیپ تزریق شده است و توسط VirtualProtect به آن دسترسی Executable ارائه شد، در شروع خود دارای فرمت MZ است که به عنوان امضای فایل‌های PE در ویندوز شناخته می‌شود.

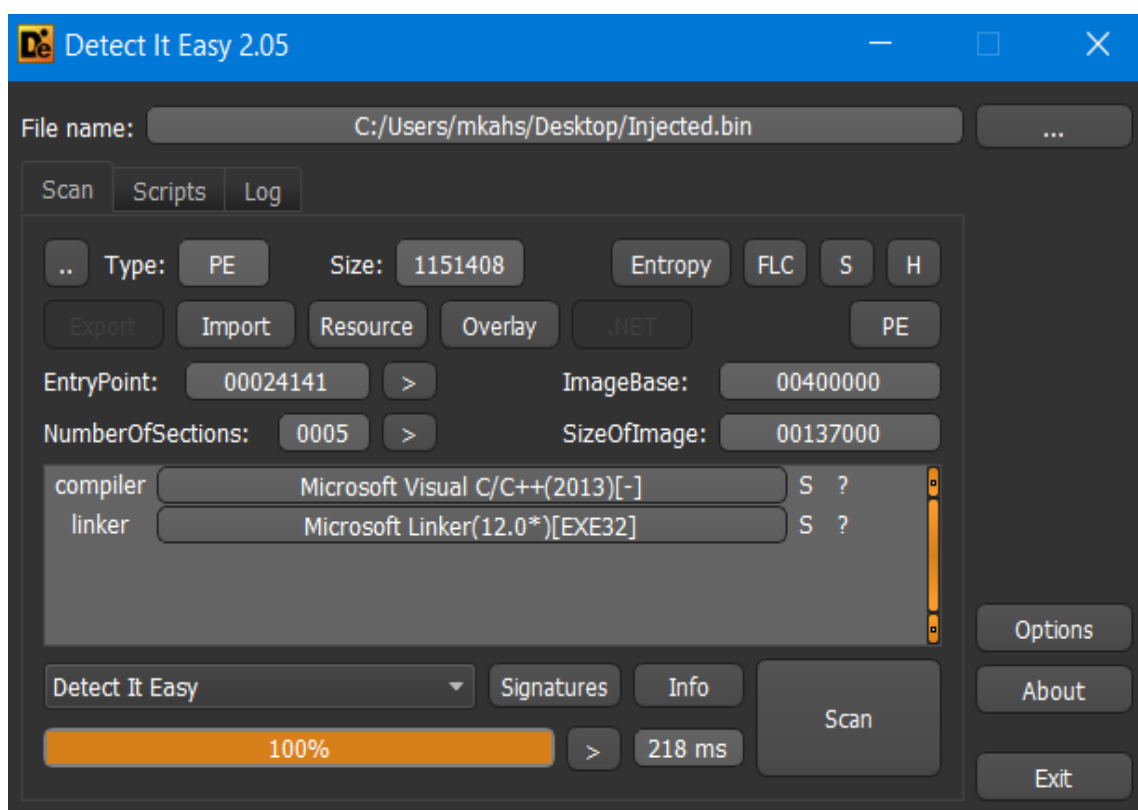
Address	Hex	ASCII
02960E50	4D 5A 90 00	MZ.....ÿÿ.
02960E60	B8 00 00 00	.....@.....
02960E70	00 00 00 00	.....
02960E80	00 00 00 00	.....
02960E90	0E 1F BA 0E	..e..!..LÍ!T
02960EA0	69 73 20 70	is program cann
02960EB0	74 20 62 65	t be run in DOS
02960EC0	6D 6F 64 65	mode...\$......
02960ED0	28 4F 86 99	(0..1.èÊ1.èÊ1.èÊ
02960EE0	D9 B0 37 CA	Ù°7Êf.èÊÙ°.Ê+ /è
02960EF0	2A 7F 09 CA	*..Êh.èÊ±N9Êm.èÊ
02960F00	F2 8E 2F CA	ò./Êm.èÊa .ÊQ.èÊ
02960F10	61 7C 37 CA	a 7Ês.èÊa .ÊØ.èÊ
02960F20	B1 D1 26 CA	±N&Ên.èÊ±N8Ên.èÊ
02960F30	B1 D1 23 CA	±N#ÊM.èÊ1.éÊ..èÊ
02960F40	D9 B0 0D CA	Ù°.Êd.èÊa 3Êm.èÊ

تصویر ۸: کد تزریق شده به حافظه هیپ توسط بدافزار

## تحلیل کد تزریق شده به حافظه هیپ

همانطور که پیش از این ذکر شد، برای ادامه تجزیه و تحلیل باج‌افزار، نیازمند هستیم از فایل تزریق شده به درون حافظه هیپ توسط بدافزار دامپ بگیریم. در این قسمت، آن را با عنوان **Injected.bin**، در قالب یک فایل اجرایی مجزا از روی حافظه دامپ گرفتیم تا سپس به تجزیه و تحلیل آن بپردازیم.

همانطور که در تصویر ۹ اطلاعات این فایل اجرایی نمایش داده شده است، مجدد با زبان **CPP** و کامپایلر **MSVC** برنامه‌نویسی و کامپایل شده است. همچنین با توجه به آنتروپی آن به نظر می‌رسد که مبهم‌سازی هم نشده باشد که این موضوع به تنهایی کار تحلیل ساختار این فایل اجرایی را برای ما ساده‌تر می‌کند.



تصویر ۹: اطلاعات فایل تزریق شده به درون هیپ توسط باج‌افزار

حال که به کد تزریق شده به درون هیپ دسترسی مجزا داریم، می‌توانیم ساختار داخلی آن را به سادگی توسط ابزارهای گوناگون مشاهده کنیم. به عنوان مثال، اگر رشته‌های درون فایل اجرایی **Injected.bin** را مشاهده کنیم، اطلاعات جالبی را به دست خواهیم آورد از قبیل اینکه در این فایل اجرایی از زیرسیستم رمزنگاری **Microsoft Enhanced RSA**، **AES Cryptographic Provider**، کتابخانه **OpenSSL** و ثابت‌های انکدینگ و دکدینگ **Netscape** و ... به

صورت گسترده استفاده شده است. در تصویر ۱۰، رشته‌هایی که گواه استفاده از ACP، MER و OpenSSL در فایل اجرایی هستند، نمایش داده شده است.

mtc	-	-	<a href="#">Montgomery Multiplication for x86_CRYPTOGAMS by &lt;appro@openssl.org&gt;</a>
-	-	-	<a href="#">Monday</a>
-	-	-	<a href="#">Monday</a>
-	-	-	<a href="#">Modulus:</a>
execution	<a href="#">Process Discovery</a>	<a href="#">Discovery</a>	<a href="#">Module32Next</a>
execution	<a href="#">Process Discovery</a>	<a href="#">Discovery</a>	<a href="#">Module32First</a>
-	-	-	<a href="#">MinghuaQuw</a>
-	-	-	<a href="#">MinghuaQuS</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">MinghuaQu</a>
-	-	-	<a href="#">Microsoft Visual C++ Runtime Library</a>
-	-	-	<a href="#">Microsoft Universal Principal Name</a>
-	-	-	<a href="#">Microsoft Trust List Signing</a>
-	-	-	<a href="#">Microsoft Smartcardlogin</a>
-	-	-	<a href="#">Microsoft Server Gated Crypto</a>
-	-	-	<a href="#">Microsoft Local Key set</a>
-	-	-	<a href="#">Microsoft Internet Explorer</a>
-	-	-	<a href="#">Microsoft Internet Explorer</a>
-	-	-	<a href="#">Microsoft Individual Code Signing</a>
-	-	-	<a href="#">Microsoft Extension Request</a>
cryptography	-	-	<a href="#">Microsoft Enhanced RSA and AES Cryptographic Provider</a>
-	-	-	<a href="#">Microsoft Encrypted File System</a>
-	-	-	<a href="#">Microsoft Commercial Code Signing</a>
-	-	-	<a href="#">Microsoft CSP Name</a>

### تصویر ۱۰: رشته‌های متعلق به فایل باینری Injected.bin

در ادامه، اگر فایل اجرایی Injected.bin را دی‌آس‌مبل کنیم و ساختار آن را مورد بررسی قرار بدهیم، مشاهده خواهیم کرد که در گام اول اجرای خود به متغیرهای محلی سیستم دسترسی می‌گیرد و مقادیری که درون بلاک محیط پروسه (Process Environment Block) وجود دارد، بررسی و پردازش می‌کند.

در مرحله بعدی، وارد روتینی خواهد شد که با وبگاه <https://api.2ip.ua/geo.json> ارتباط خواهد گرفت، تا اطلاعات ماشین قربانی از قبیل آدرس IP، کد کشور، نام کشور، نام شهر، منطقه، مختصات جغرافیایی و ... را استخراج کند. در تصویر ۱۱، اطلاعاتی بازگشتی توسط این وبسایت هنگام برقراری ارتباط با آن نمایش داده شده است.

```

{"ip":"46.167.135.182","country_code":"IR","country":"Iran (islamic
republic
of)","country_rus":"\u0418\u0440\u0430\u0433\u043e\u043d","region":"Golestan","
region_rus":"\u0413\u043e\u043b\u0435\u0441\u0442\u0430\u043d","cit
y":"Gorgan","city_rus":"\u0413\u043e\u0440\u0433\u043e\u0433\u043e\u0433\u0434","lati
tude":"36.83866","longitude":"54.43475","zip_code":"-
","time_zone":"+04:30"}

```

تصویر ۱۱: اطلاعات بازگشتی توسط سایت 2ip.ua

در تصویر ۱۲، قسمتی از کد فایل اجرایی Injected.bin را در محیط دیباگر مشاهده می‌کنید که بدافزار با استفاده از رابط برنامه‌نویسی InternetOpenUrl تلاش به برقراری ارتباط با وبسایت 2ip.ua و خواندن اطلاعات بازگشتی توسط این وبسایت در قالب یک فایل JSON کرده است.

EIP	Address	Instruction	Comment
730D53F0	8BFF	mov edi,edi	InternetOpenUrlA
730D53F2	55	push ebp	
730D53F3	8BEC	mov ebp,esp	
730D53F5	83EC 3C	sub esp,3C	
730D53F8	8D45 C4	lea eax,dword ptr ss:[ebp-3C]	
730D53FB	56	push esi	esi:"https://api.2ip.ua/geo.json"
730D53FC	6A 3C	push 3C	
730D53FE	6A 00	push 0	
730D5400	50	push eax	
730D5401	E8 9741FAFF	call <JMP.&memset>	
730D5406	83C4 0C	add esp,C	
730D5409	8D4D C4	lea ecx,dword ptr ss:[ebp-3C]	
730D540C	E8 8F78F2FF	call wininet.72FFCCA0	
730D5411	FF75 1C	push dword ptr ss:[ebp+1C]	
730D5414	8B55 0C	mov edx,dword ptr ss:[ebp+C]	[ebp+C]:L"https://api.2ip.ua/geo.json"
730D5417	FF75 18	push dword ptr ss:[ebp+18]	
730D541A	8B4D 08	mov ecx,dword ptr ss:[ebp+8]	
730D541D	FF75 14	push dword ptr ss:[ebp+14]	
730D5420	FF75 10	push dword ptr ss:[ebp+10]	
730D5423	E8 D6F3FFFF	call wininet.730D47FE	
730D5428	8D4D C4	lea ecx,dword ptr ss:[ebp-3C]	
730D542B	8BF0	mov esi,eax	esi:"https://api.2ip.ua/geo.json"
730D542D	E8 BC76F2FF	call wininet.72FFCAEE	
730D5432	8BC6	mov eax,esi	esi:"https://api.2ip.ua/geo.json"
730D5434	5E	pop esi	esi:"https://api.2ip.ua/geo.json"
730D5435	C9	leave	

تصویر ۱۲: برقراری ارتباط با 2ip

در تصویر ۱۳، فراخوانی تابع `HttpQueryInfoW` را مشاهده می‌کنیم که بدافزار با استفاده از این رابط برنامه‌نویسی اطلاعات بازگشتی توسط وبسایت `zip.ua` را می‌خواند و پردازش می‌کند. وقتی وبسایت `zip.ua` اطلاعات ماشین‌کاربر را بازگشت داد، بدافزار با بررسی کد کشور قربانی کار خود را ادامه می‌دهد.

```

loc_41E079:          ; lpszUrl
push    eax
push    edi          ; hInternet
call   ds:InternetOpenUrlA
cmp    [esp+2930h+var_2860], 10h
mov    edi, eax
jb     short loc_41E09C

[esp+2930h+var_2874]; void *
call   j__free
add    esp, 4

loc_41E09C:
mov    [esp+2930h+var_2860], 0Fh
mov    [esp+2930h+var_2864], 0
mov    byte ptr [esp+2930h+var_2874], 0
test   edi, edi
jnz   short loc_41E0E2

loc_41E0E2:          ; lpdwIndex
push    0
lea    eax, [esp+2934h+dwBufferLength]
mov    [esp+2934h+Buffer], 0
push    eax          ; lpdwBufferLength
lea    eax, [esp+2938h+Buffer]
mov    [esp+2938h+dwBufferLength], 4
push    eax          ; lpBuffer
push    20000013h    ; dwInfoLevel
push    edi          ; hRequest
call   ds:HttpQueryInfoW
cmp    [esp+2930h+Buffer], 12Ch
jnb   short loc_41E0BE
    
```

تصویر ۱۳: استفاده از `HttpQueryInfoW`

در تصویر ۱۴، این مسئله نمایش داده شده است. همینطور که در این تصویر مشاهده می‌کنید، بعد اینکه اطلاعات بازگشتی توسط بدافزار خوانده می‌شود، باج افزار به تابع `1656D0` وارد خواهد شد تا این اطلاعات را بررسی دقیق کند.

0015CFB4	E9 5A020000	jmp injected.15D213	
0015CFB9	8D85 68FFFFFF	lea eax,dword ptr ss:[ebp-98]	
0015CFBF	50	push eax	
0015CFC0	68 00280000	push 2800	
0015CFC5	8D85 68D7FFFF	lea eax,dword ptr ss:[ebp-2898]	
0015CFCB	50	push eax	
0015CFCC	56	push esi	
0015CFCD	FF15 A8C32100	call dword ptr ds:[&InternetReadFile]	
0015CFD3	56	push esi	
0015CFD4	8B35 A4C32100	mov esi,dword ptr ds:[&InternetCloseHandle]	
0015CFDA	FFD6	call esi	
0015CFDC	57	push edi	
0015CFDD	FFD6	call esi	
0015CFDF	6A 10	push 10	
0015CFE1	68 ECF24000	push injected.24FFEC	24FFEC:"\country_code\":"

تصویر ۱۴: عبور کد کشور قربانی به یک روتین دیگر

در تصویر ۱۵، نحوه پردازش اطلاعات را مشاهده می کنید. همانطور که در این تصور نمایش داده شده است، اطلاعات بازگشتی توسط وب سایت Zip.ua توسط بدافزار بایت به بایت مورد پردازش قرار می گیرد تا در ادامه بتواند با استفاده از این اطلاعات به شیوه دقیق تری با قربانی تعامل برقرار و از آن باج خواهی کند.

0015D021	EB 16	jmp injected.15D039	
0015D023	8D8D 68D7FFFF	lea ecx,dword ptr ss:[ebp-2898]	
0015D029	8D51 01	lea edx,dword ptr ds:[ecx+1]	edx:"\ip":"46.167.135.182","\country_code":"IR","\country
0015D02C	8D6424 00	lea esp,dword ptr ss:[esp]	[esp]:EntryPoint
0015D030	8A01	mov al,byte ptr ds:[ecx]	ecx:"46.167.135.182","\country_code":"IR","\country":"Ira
0015D032	41	inc ecx	ecx:"46.167.135.182","\country_code":"IR","\country":"Ira
0015D033	84C0	test al,al	
0015D035	75 F9	jne injected.15D030	
0015D037	2BCA	sub ecx,edx	ecx:"46.167.135.182","\country_code":"IR","\country":"Ira
0015D039	51	push ecx	ecx:"46.167.135.182","\country_code":"IR","\country":"Ira
0015D03A	8D85 68D7FFFF	lea eax,dword ptr ss:[ebp-2898]	

### تصویر ۱۵: پردازش اطلاعات بازگشتی توسط Zip.ua

در گام بعد کد کشور قربانی با کدهای کشوری که در بدافزار نهفته شده است، مقایسه خواهد شد تا اگر قربانی دارای کد کشوری مشابه با لیست نهفته شده در بدافزار بود، نوع عملکرد باج افزار شخصی سازی و هدفمندتر شود. در تصویر زیر، کادر قرمز رنگ، کد کشورهایی است که در بدافزار نهفته شده اند و مقدار IR کد کشور قربانی (ماشین تحلیلگر بدافزار) است.

0015D105	C785 70FFFFFF	mov dword ptr ss:[ebp-90],injected.250008	[ebp-90]:"BY", 250008:"BY"
0015D10F	C785 74FFFFFF	mov dword ptr ss:[ebp-8C],injected.25000C	[ebp-8C]:"UA", 25000C:"UA"
0015D123	C785 78FFFFFF	mov dword ptr ss:[ebp-88],injected.250010	[ebp-88]:"AZ", 250010:"AZ"
0015D12D	C785 7CFFFFFF	mov dword ptr ss:[ebp-84],injected.250014	[ebp-84]:"AM", 250014:"AM"
0015D134	C745 80 1800250	mov dword ptr ss:[ebp-80],injected.250018	[ebp-80]:"TJ", 250018:"TJ"
0015D13B	C745 84 1C00250	mov dword ptr ss:[ebp-7C],injected.25001C	[ebp-7C]:"KZ", 25001C:"KZ"
0015D142	C745 88 2000250	mov dword ptr ss:[ebp-78],injected.250020	[ebp-78]:"KG", 250020:"KG"
0015D149	C745 8C 2400250	mov dword ptr ss:[ebp-74],injected.250024	[ebp-74]:"UZ", 250024:"UZ"
0015D150	C745 90 2800250	mov dword ptr ss:[ebp-70],injected.250028	[ebp-70]:"SY", 250028:"SY"
0015D157	8B94BD 6CFFFFFF	mov edx,dword ptr ss:[ebp+edi*4-94]	[ebp+edi*4-94]:"RU"
0015D15A	803A 00	cmp byte ptr ds:[edx],0	edx:"RU"
0015D15C	75 04	jne injected.15D160	
0015D15E	33F6	xor esi,esi	esi:&"AM"
0015D160	EB 0E	jmp injected.15D16E	
0015D162	8BF2	mov esi,edx	esi:&"AM", edx:"RU"
0015D165	8D4E 01	lea ecx,dword ptr ds:[esi+1]	esi:&"AM"
0015D167	8A06	mov al,byte ptr ds:[esi]	esi:&"AM"
0015D16A	46	inc esi	esi:&"AM"
0015D16C	84C0	test al,al	
0015D16E	75 F9	jne injected.15D165	
0015D172	2BF1	sub esi,ecx	esi:&"AM"
0015D175	837D C0 10	cmp dword ptr ss:[ebp-40],10	
0015D177	8D4D AC	lea ecx,dword ptr ss:[ebp-54]	
0015D17B	8BC6	mov eax,esi	eax:"IR", esi:&"AM"
0015D17D	0F434D AC	cmovae ecx,dword ptr ss:[ebp-54]	
0015D17F	3BDE	cmp ebx,esi	esi:&"AM"

### تصویر ۱۶: کد کشورهای نهفته شده در باج افزار

در گام بعد، فایل اجرایی Injected.bin برای اینکه با موفقیت بتواند عملیات های مورد نظر خود را انجام بدهد، رابط برنامه نویسی ShellExecute را به همراه پارامتر runas اجرا خواهد کرد تا دسترسی خود را افزایش دهد. در تعریف این رابط برنامه نویسی در مایکروسافت آورده شده است که این رابط برنامه نویسی هنگامیکه با پارامتر runas اجرا می شود، یک برنامه کاربردی را با سطح دسترسی Administrator اجرا خواهد کرد. در تصویر ۱۷، فرخوانی ShellExecute توسط بدافزار نمایش داده شده است.

```

0016A690 8D8424 E8030000 lea eax,dword ptr ss:[esp+3E8]
0016A697 C78424 B0010000 mov dword ptr ss:[esp+180],0
0016A6A2 898424 C0010000 mov dword ptr ss:[esp+1C0],eax
0016A6A9 8D46 10 lea eax,dword ptr ds:[esi+10]
0016A6AC 8378 14 08 cmp dword ptr ds:[eax+14],8
0016A6B0 - 72 02 jb injected.16A6B4
0016A6B2 8B00 mov eax,dword ptr ds:[eax]
0016A6B4 898424 C4010000 mov dword ptr ss:[esp+1C4],eax
0016A6BB 8D8424 AC010000 lea eax,dword ptr ss:[esp+1AC]
0016A6C2 50 push eax
0016A6C3 C78424 CC010000 mov dword ptr ss:[esp+1CC],5
0016A6CE C78424 BC010000 mov dword ptr ss:[esp+1BC],injected.253FF0
0016A6D9 FF15 20C32100 call dword ptr ds:[<&ShellExecuteExW>]
EIP -> 0016A6DF 85C0 test eax,eax
0016A6E1 - 74 1B je injected.16A6FE
0016A6E3 33F6 xor esi,esi

```

dword ptr [esp+1BC]=[006FE774 &L"C:\Users\mkahs\Desktop\Malware\Ransomware\Injected.bin"]=006FF288 L"C:\Users\mkahs\Desktop\Malware\Ransomware\Injected.bin" 00253FF0 L"runas"

.text:0016A6CE injected.bin:\$1A6CE #19ACE

### تصویر ۱۷: اجرای پیلود اجرایی بدافزار توسط رابط ShellExecute

سپس بدافزار تلاش خواهد کرد از یک وب سایت با آدرس [www.nokd.top](http://www.nokd.top) چند فایل اجرایی دانلود و در مسیر AppData ذخیره سازی کند. این فایل ها با نام های updatewin.exe، updatewin1.exe، updatewin2.exe، 3.exe و 5.exe هستند که به دلیل غیرفعال بودن وبسایت nokd.top این عملیات با موفقیت همراه نخواهد شد و تمامی فایل هایی که بر روی مسیر AppData ایجاد می شوند، خراب خواهند بود. در تصویر ۱۸، فایل هایی که توسط بدافزار بر روی مسیر AppData قرار گرفته اند، نمایش داده شده است.

This PC > Local Disk (C:) > Users > mkahs > AppData > Local > 5620b487-e9e8-43c5-8291-d9f7bcf0d2c7

Name	Date modified	Type	Size
3.exe	5/14/2020 9:20 AM	Application	0 KB
updatewin.exe	5/14/2020 9:19 AM	Application	0 KB
updatewin1.exe	5/14/2020 9:17 AM	Application	0 KB
updatewin2.exe	5/14/2020 9:18 AM	Application	0 KB

### تصویر ۱۸: فایل های مرتبط با بدافزار Injected.bin

سپس برای شروع رمزنگاری کلید عمومی (Public Key) را ایجاد می کند و سپس بایت به بایت آن را مورد بررسی قرار می دهد. در تصویر ۱۹ مقدار کلید عمومی را مشاهده می کنید که در یک حلقه بایت به بایت آن مورد بررسی قرار می گیرد.



### تصویر ۱۹: کلید عمومی ایجاد شده توسط بدافزار

کلید عمومی که توسط بدافزار ایجاد شده است، در زیر آورده شده است. بعد ایجاد این کلید عمومی، در هر دیرکتوری که بدافزار عمل رمزنگاری انجام می دهد، یک فایل با نام `_readme.txt` ایجاد خواهد شد که در برگزیده اطلاعات تماس به توسعه دهندگان بدافزار و شناسه ماشین قربانی و ... است.

-----BEGIN PUBLIC KEY-----

```

\\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAA6mc4av0qBgVc00vu
120E\\n6TPCO04Qbf79EbDUdCP3FO5jFuLLmOa15oplHveKsXedyzhurGAunV6Wj
hjWxGxK\\nD43w210FT8uH\\zZPqDvnatctgugSZHxRLMhBldi6JMz6hrYVvYRDXL1
y7juR9ml9w\\ncxcTDUr19L32HLpsSzLXqIa\\Vfg9MuNtfWnGvWtOs19w5Oo5fju
T\\PVhSZ5k65F6X\\n579S3wEiV3YoC3PesK2JEHLH60LEOPqIwaHbvIDGODi3U2
hDIYNENZFSFyS5dqaX\\nFEFdOpEqya8Nh77OEIK91fCWYXmK7I2UmZwPOtLihST
4iweNNMRmaw8p7bglLHbj\\n5QIDAQAB\\n

```

-----END PUBLIC KEY-----

در نهایت وقتی بدافزار تمامی اطلاعات مربوط به فایل ها و دیرکتوری ها را به دست آورد، فرآیند رمزنگاری با استفاده از توابعی از قبیل `CryptAcquireContext`، `CryptStringToBinary`، `CryptEncrypt` و ... را شروع می کند. در تصویر ۲۰، روتینی که عمل رمزنگاری را انجام می دهد، آورده شده است.

در ابتدا `CryptAcquireContext` فرخوانی می شود تا یک هندل به `CSP` به دست آورد. شایان ذکر است، سرویس `CSP` یا ارائه دهنده سرویس رمزنگاری میکروسافت (`Cryptographic Service Provider`) یک کتابخانه نرم افزاری است که رابط های رمزنگاری میکروسافت (`Microsoft CryptoAPI`) را پیاده سازی کرده است. این کتابخانه به توسعه دهندگان نرم افزار و بدافزار اجازه می دهد تا داده های خود را رمزنگاری و رمزگشایی کنند.

به هر صورت، همانطور که در تصویر ۲۰ قابل مشاهده است، بدافزار در گام اول یک آبجکت هش (Hash Object) از نوع MD5 ایجاد می‌کند. بعد هش MD5 مقدار درون متغیر V3 را محاسبه خواهد کرد (متغیر V4 همچنین طول متغیر V3 است). بعد مقدار هش ایجاد شده را با CryptGetHashParam می‌خواند و در متغیر v5 قرار می‌دهد.

```

26 v15 = 0;
27 sub_4156D0(a3, &unk_4FFCA4, 0);
28 v20 = 0;
29 if ( !CryptAcquireContextW(&phProv, 0, 0, 1u, CRYPT_VERIFYCONTEXT) )
30 {
31     v14 = 0;
32     _CxxThrowException(&v14, &_TI1H);
33 }
34 if ( !CryptCreateHash(phProv, 0x8003u, 0, 0, &phHash) )
35 {
36     v13 = 0;
37     _CxxThrowException(&v13, &_TI1H);
38 }
39 if ( !CryptHashData(phHash, v3, v4, 0) )
40 {
41     v12 = 0;
42     _CxxThrowException(&v12, &_TI1H);
43 }
44 pdwDataLen = 0;
45 if ( !CryptGetHashParam(phHash, 2u, 0, &pdwDataLen, 0) )
46 {
47     v11 = 0;
48     _CxxThrowException(&v11, &_TI1H);
49 }
50 v5 = (unsigned __int8 *)operator new[](pdwDataLen + 1);
51 v15 = v5;
52 memset(v5, 0, pdwDataLen + 1);
53 if ( !CryptGetHashParam(phHash, 2u, v5, &pdwDataLen, 0) )
54 {
55     v10 = 0;
56     _CxxThrowException(&v10, &_TI1H);
57 }
58 for ( i = 0; i < pdwDataLen; ++i )
59 {
60     sprintf(&v9, "%.2X", v5[i]);
61     if ( v9 )
62         sub_413EA0(&v9, strlen(&v9));
63     else
64         sub_413EA0(&v9, 0);
65 }
66 jz__free(v5);
67 CryptDestroyHash(phHash);
68 CryptReleaseContext(phProv, 0);
69 return 1;

```

### تصویر ۲۰: روتین رمزنگاری استریم‌های داده‌ای باج‌افزار

در نهایت اگر مقدار هش که محاسبه و خوانده شده است، دارای خطا یا نامعتبر باشد، یک exception ایجاد می‌شود. در غیر این صورت، هر بایت هش به معادل رشته‌ای خود تبدیل می‌شود، و اگر معادل رشته‌ای معتبر بود (یعنی مخالف NULL بود)، تابع sub\_413EA0 صدا زده خواهد شد و معادل رشته‌ای، به عنوان آرگومان به آن ارائه می‌شود. این روال برای تمام بایت‌های هش تکرار می‌شود. در آخر هم آبجکت هش پس از این که از حلقه for بیرون آمدیم، از روی حافظه پاک و آزاد خواهد شد.

# نتیجه گیری

باج افزار Speedy که از خانواده STOP/DJVVU است، از هر نظر برای مبحث امنیت ارتباطات کشور جمهوری اسلامی ایران اهمیت راهبردی دارد، زیرا این باج افزار، توانسته است با گستردگی بسیار زیادی کاربران و سازمان های ایرانی را آلوده کند.

اگر چه به نظر می رسد، نسخه فعلی که از این باج افزار انتشار پیدا کرده است، آخرین نسخه از این باج افزار نباشد و در آینده نسخه های عملیاتی و خاص این بدافزار مجدد کاربران و زیرساخت فناوری اطلاعات ایران را هدف قرار بدهد. از همین روی، پیش از اینکه سامانه های زیرساختی کشور مورد حمله این بدافزار قرار بگیرند، باید گام هایی به منظور حفظ یکپارچگی و عملکرد آن ها برداشته شود.

## نشانه نفوذگر «IOC»

Name:	Hash
Samples:	E93A3705FDD01CF942E4830E148F0B66
	4F640DA558A758DD5D58483C150E6D41A05F3766
	DCE16104C90479656ECEA5C63B5C4A3B
	ACFC8C08A67E915E0187713CC0912AEFF00560A5
URLs	<a href="http://nokd.top">http://nokd.top</a>
	<a href="https://api.2ip.ua/geo.json">https://api.2ip.ua/geo.json</a>
	<a href="http://www.openssl.org/support/faq.html">http://www.openssl.org/support/faq.html</a>

جدول ۳: نشان نفوذ «IOC»